

Dipl.-Ing. Werner Dieterle, Mannheim

**Effiziente Kommunikations-
Architekturen zum Aufbau
verteilter Echtzeit-Daten-
banken in industriellen
Leitsystemen**

Reihe **10**: Informatik/
Kommunikationstechnik Nr. **436**

1	Einführung	1
1.1	Motivation	1
1.2	Übersicht	2
2	Architektur moderner Leitsysteme	4
2.1	Basisarchitektur	4
2.1.1	Hard- und Softwarearchitektur	4
2.1.2	Datenmodelle und Datenflüsse	5
2.1.3	MMI-Bildkonzepte	6
2.2	Anforderungen an Leitsysteme	7
2.3	Spezifische Architekturmerkmale	9
2.3.1	Offenheit	9
2.3.2	Verteilte Anwendungsplattform	10
2.3.3	Funktionsrechnerkonzept	10
2.3.4	Verteilte Datenbank	11
2.3.5	Producer/Consumer-Kommunikation	14
2.3.6	Fehlertoleranz	14
2.4	Grundzüge des Aufbaus fehlertoleranter Systeme	15
2.4.1	Fehler- und Ausfallmodell der Komponenten	15
2.4.1.1	Fehler- und Ausfallursachen	15
2.4.1.2	Hardwarekomponenten eines verteilten Systems	16
2.4.1.3	Fehlerverhalten der Komponenten	18
2.4.2	Realisierung struktureller Redundanz	22
2.4.2.1	Aufbau redundanter Rechnersysteme	23
2.4.2.2	Aufbau redundanter LAN-Systeme	24
2.4.2.3	Weitere Maßnahmen der Zuverlässigkeitserhöhung	24
2.5	Zusammenfassung	25
3	Probleme und Grundsätze redundanter Datenmodelle	26
3.1	Warum redundante Datenmodelle?	26
3.2	Vergleich mit konventionellen („verteilten“) Systemen	27
3.3	Existierende Lösungen verteilter Datenbanken	29
3.4	Datenkonsistenz	30
3.5	Grundsätze datenkonsistenten Nachrichtenaustauschs	32
3.5.1	Nachrichtenintegrität	33
3.5.2	Synchronisation konkurrierender Informationsquellen	34
3.5.2.1	Exklusiver Zugriff	36
3.5.2.2	Erweiterte totale Reihenfolge	36

3.5.2.3	Führungskompetenz	36
3.5.3	FIFO-Reihenfolge	37
3.5.4	Führungskonsistenz	38
3.6	Grundsätze zur Nachrichtenreihenfolge	41
3.6.1	Totale Reihenfolge (Reihenfolge-Konsistenz)	41
3.6.2	Kausale Reihenfolge	42
3.7	Weitere Grundsätze des Nachrichtentransports	43
3.7.1	Konsistente Systemsicht mit Fehlersignalisierung	43
3.7.2	Unterstützung von Rechnerredundanz	43
3.7.3	Funktionsrechnerkonzept	44
3.7.4	Nahtstellensystem	44
3.7.5	Empfängeranonyme Übertragung	45
3.8	Protokoll-Architektur datenkonsistenter Kommunikation	46
3.9	Vergleich mit existierenden Lösungen verteilter Datenbanken	48
3.10	Formale Betrachtung zur Konsistenz	49
3.10.1	Hinreichendes und notwendiges Kriterium der Datenkonsistenz	49
3.10.2	Allgemeiner Konsistenz-Begriff	50
3.11	Zusammenfassung	52
4	Bausteine der Führungskonsistenz	53
4.1	Existierende Konzepte	53
4.2	Annahmen/Voraussetzungen	56
4.3	Definition von Verfahrensklassen	56
4.4	Verfahren mit Vermeidung von Inkonsistenzen im Fehlerfall	58
4.4.1	Fehlertoleranz-Phasen	58
4.4.2	Führungskonsistenz und Rücksetzpunkterstellung	58
4.4.3	Normalbetrieb und Fehlererkennung	60
4.4.3.1	Führungskonsistenz aus Teilnehmersicht	60
4.4.3.2	Basiskonzept	62
4.4.3.3	Optimierte spontane Verfahren (Signalisierung des Erhalts)	63
4.4.3.4	Optimierte zyklische Verfahren (Signalisierung des Nicht-erhalts)	65
4.4.3.5	Berücksichtigung von Übertragungsfehlern, Bus-, Controller- und Empfängerausfällen	68
4.4.3.6	Übertragungsaufwand	69
4.4.4	Fehlerdiagnose	70
4.4.5	Fehlerbehandlung	71
4.4.5.1	Rekonfiguration	71
4.4.5.2	Restauration der Führungskonsistenz	72
4.4.6	Tolerierung von Mehrfachfehlern	73

4.4.7	Fazit	74
4.5	Verfahren ohne spezifische Maßnahmen im Fehlerfall (Nonstop-Verfahren)	76
4.6	Verfahren mit Korrektur von Inkonsistenzen im Fehlerfall	78
4.7	Zustandsorientierte Verfahren	79
4.8	Zusammenfassung	79
5	Bausteine konsistenter Reihenfolge und Systemsicht	81
5.1	Konsistente (totale) Reihenfolge von Nachrichten	81
5.1.1	FIFO-Reihenfolge	81
5.1.2	Totale Reihenfolge aus Teilnehmersicht	82
5.1.3	Lösungsansätze zur totalen Reihenfolge	83
5.1.3.1	Führungskompetenz	83
5.1.3.2	Sequencer	84
5.1.3.3	Deterministischer Buszugriff mit führungskonsistenter Übertragung	85
5.1.3.4	Stochastischer Buszugriff mit führungskonsistentem physikalischem Broadcast	86
5.1.3.5	Zyklische Übertragung	87
5.1.3.6	Verwendung synchronisierter Uhren	88
5.2	Kausale Reihenfolge von Nachrichten	88
5.3	Synchronisation konkurrierender Informationsquellen	90
5.4	Konsistente Systemsicht	90
5.5	Zusammenfassung	92
6	Verwendung standardisierter Protokolle unter UNIX	93
6.1	Fehlertolerante UNIX-Systeme	93
6.2	Standardisierte Protokolle: Einführende Beschreibung	94
6.2.1	Das Client/Server-Modell	94
6.2.2	Client/Server-Protokolle: RPC und NFS	96
6.2.3	Basisprotokolle: LAN, IP, ICMP	96
6.2.4	Höhere Protokolle: TCP und UDP	97
6.2.5	Socket- und TLI-Schnittstelle	99
6.2.6	Server-Basistypen	99
6.3	Herkömmliche Verwendung standardisierter Protokolle	100
6.3.1	Fehlererkennung	102
6.3.2	Weitere Probleme im Normalbetrieb	103
6.3.3	Fehlerdiagnose und -behandlung	105

6.4	Zusammenfassung	106
7	Angepaßte Verfahren zur Kommunikation in Leitsystemen	107
7.1	Eingrenzung von Lösungen	107
7.2	Ring-Multicast	109
7.2.1	Verfahrensablauf im Normalbetrieb	109
7.2.2	Fehlertoleranz-Mechanismen	111
7.2.3	Realisierung	116
7.2.4	Experimentelle Ergebnisse	117
7.2.5	Vergleich mit existierenden Token-Protokollen	119
7.3	Datagramm-Multicast	119
7.3.1	Verfahrensablauf im Normalbetrieb	120
7.3.2	Fehlertoleranz-Mechanismen	123
7.3.3	Realisierung	123
7.3.4	Experimentelle Ergebnisse	123
7.3.5	Vergleich mit existierenden Verfahren	125
7.3.6	Verfahrensvariante für beliebige Mehrfachfehler	125
7.3.7	Weitere Eigenschaften (Ring-Multicast und Datagramm-Multicast)	127
7.4	2-Phase-Repeat	128
7.4.1	Verfahrensbeschreibung	128
7.4.2	Erweiterung auf totale Reihenfolge und Führungskompetenz	132
7.4.3	Realisierung mit Datagramm-Multicast	132
7.4.4	Theoretische und experimentelle Ergebnisse	133
7.5	Zusammenfassung	134
8	Zusammenfassung und Ausblick	135
Anhang		138
A.1	Hinreichendes und notwendiges Kriterium der Datenkonsistenz	138
A.2	Fehlertoleranz-Maßnahmen bei gerichteter Übertragung mit positiver Empfangsbestätigung	140
A.3	Nachweis minimaler Kommunikationslast des Verfahrens Datagramm-Multicast	144
Literatur		154